Class 07: Kernel-based learning methods

Martin Slawski

Volgenau School of Engineering
Department of Statistics

March 22, 2018

Kernel-based learning methods

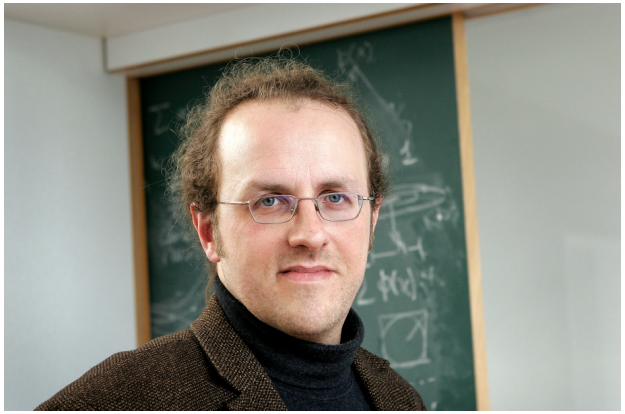
- a paradigm mainly developed in the late 1990s to $\sim$2005
- the mathematical foundations were laid out much earlier (1950s)
- first application in statistics: spline smoothing

Kernel methods offer a principled way for addressing the following two problems:

- learning non-linear functions,
- machine learning for non-vectorial data (e.g. strings, graphs, etc.)

Kernel-based learning methods

Pioneers of the approach:



Bernhard Schölkopf
MPI for Intelligent Systems

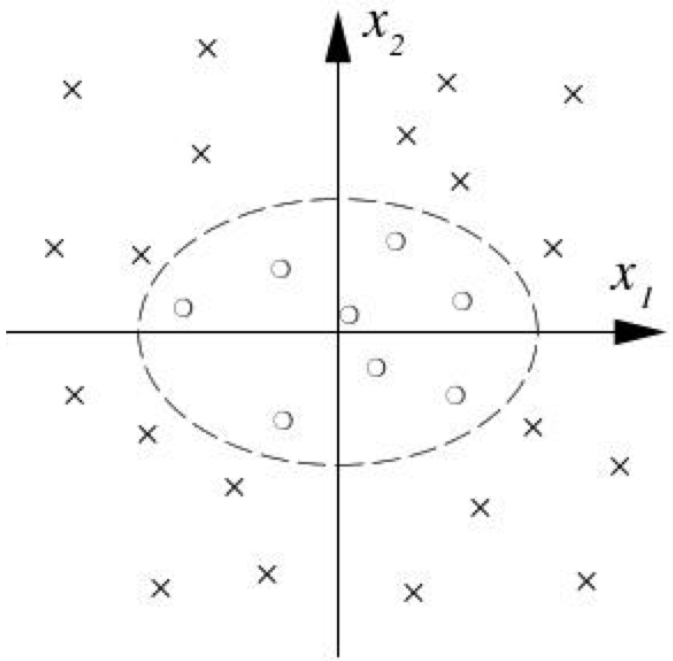
Alex Smola
AWS

B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT press, 2001.* Available for online reading via Mason's library.

http://www.kernel-machines.org/tutorials

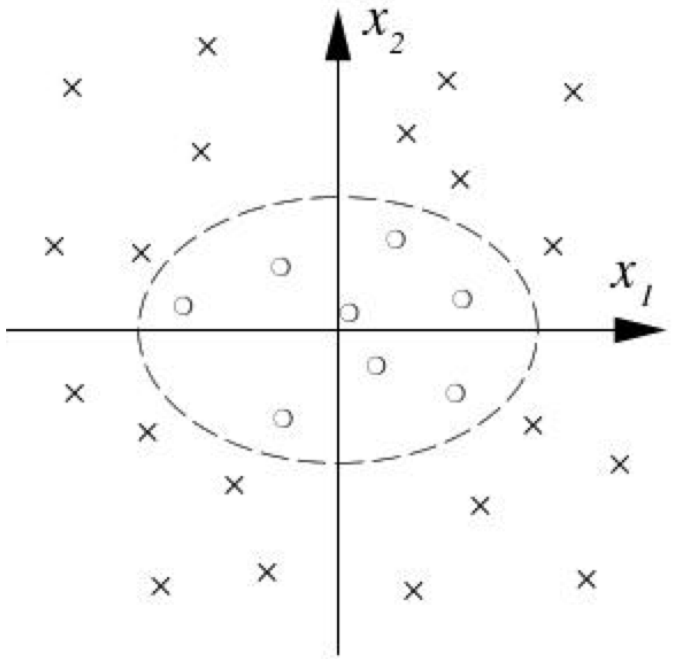
Kernel-based learning methods

Consider the following binary classification problem (Fig. 2.1 in the book cited above):



A linear classifier is not suitable in this situation.

Kernel-based learning methods



The decision boundary for the problem is given by an ellipse, i.e. by a set of the form

$$\mathcal{E} = \{(x_1, x_2) \in \mathbb{R}^2 : \; x_1^2 q_{11}^2 + x_2^2 q_{22}^2 + 2 q_{12} x_1 x_2 = c^2\}$$

Kernel-based learning methods

The decision boundary for the problem is given by an ellipse, i.e. by a set of the form

$$\mathcal{E} = \{(x_1, x_2) \in \mathbb{R}^2 : \; x_1^2 q_{11}^2 + x_2^2 q_{22}^2 + 2 q_{12} x_1 x_2 = c^2\}$$

Equivalently,

$$\mathcal{E} = \{(x_1, x_2) \in \mathbb{R}^2 : \; f(x) = 0\},$$

where

$$f(x) = x_1^2 q_{11}^2 + x_2^2 q_{22}^2 + 2 q_{12} x_1 x_2 - c^2,$$

so that x belongs to one of the two classes depending on $\text{sign}(f(x))$:

- < 0: inside the ellipse,
- > 0: outside the ellipse.

Kernel-based learning methods

Consider the feature map

$$\begin{array}{rccc} \Phi: & \mathbb{R}^2 & \to & \mathbb{R}^3 \\ & x & \mapsto & \Phi(x) = \begin{pmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix} \end{array}$$

Then we can describe $\mathcal{E}$ as a hyperplane in $\texttt{range}(\Phi)$:

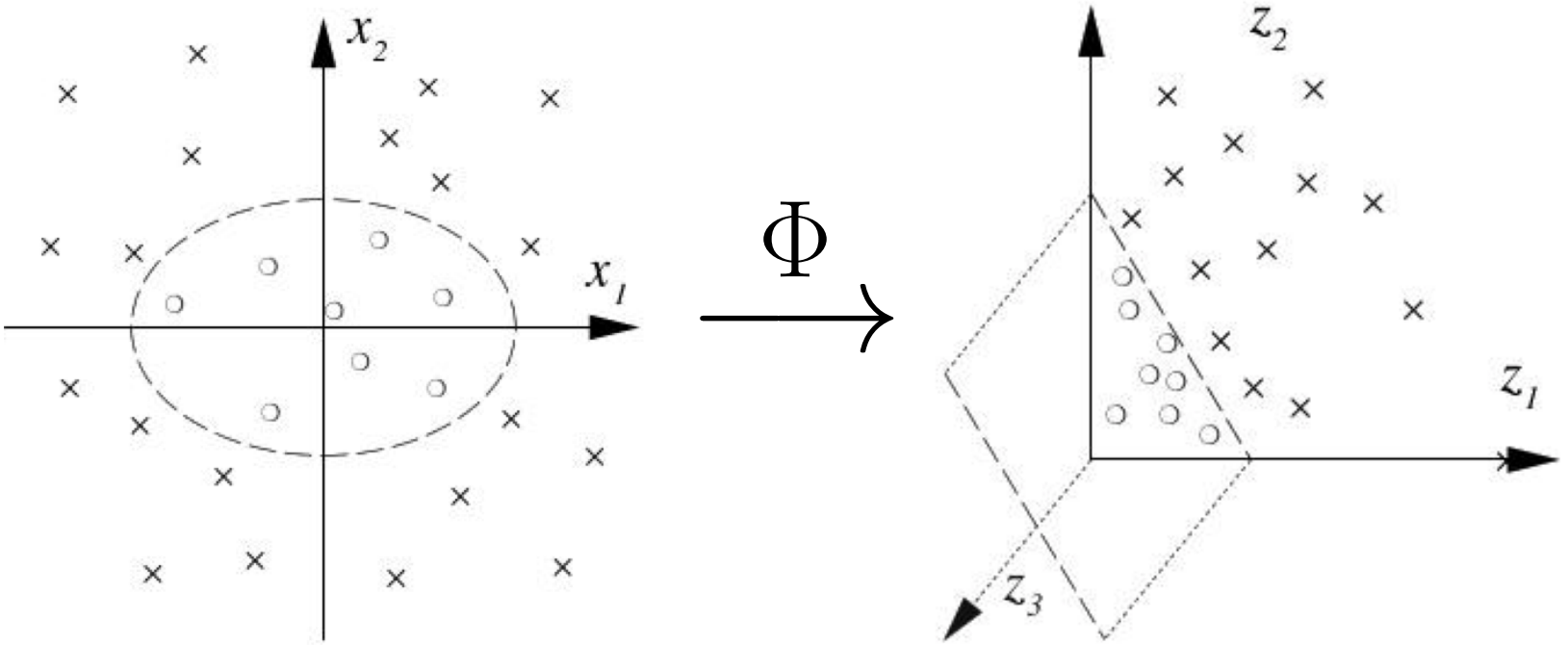
$$\begin{aligned} \mathcal{E} &= \{x: \ x_1^2q_{11}^2 + x_2^2q_{22}^2 + 2q_{12}x_1x_2 - c^2 = 0\} \\ &= \{x: \ w_0 + \langle \Phi(x), w\rangle = 0\}, \end{aligned}$$

where

$$w_0 = -c^2, \ \ w = \begin{pmatrix} q_{11}^2 \\ q_{22}^2 \\ \sqrt{2}q_{12} \end{pmatrix}$$

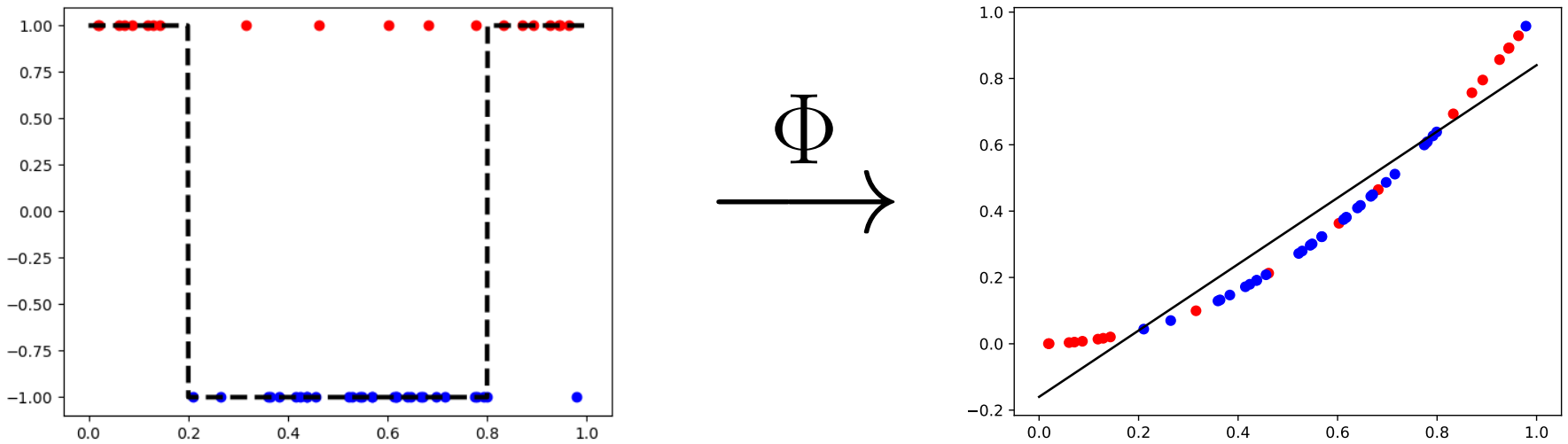
Kernel-based learning methods

Visualization of the concept:



Kernel-based learning methods

Recall the toy problem from HW 2 ($\mathcal{X} = [0, 1]$). We use $\Phi(x) = (x \quad x^2)^\top$:



Right plot, solid black line: optimal separating hyperplane.

Kernel-based learning methods

After applying the map Φ, we can train a SVM to learn a classifier that is non-linear in the original data domain.

Recall that the dual optimization problem of the soft margin SVM is given by:

$$\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \alpha_{i'} Y_i Y_{i'} \langle \Phi(X_i), \Phi(X_{i'}) \rangle$$

$$\text{subject to } \sum_{i=1}^{n} \alpha_i Y_i = 0$$

$$0 \leq \alpha_i \leq \frac{C}{n}, \;\; i = 1, \ldots, n.$$

Kernel-based learning methods

Let us inspect the objective in a bit more of detail:

$$\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \alpha_{i'} Y_i Y_{i'} \langle \Phi(X_i), \Phi(X_{i'}) \rangle$$

We have

$$\begin{aligned}
\langle \Phi(X_i), \Phi(X_{i'}) \rangle &= \left\langle \begin{pmatrix} X_{i1}^2 \\ X_{i2}^2 \\ \sqrt{2} X_{i1} X_{i2} \end{pmatrix}, \begin{pmatrix} X_{i'1}^2 \\ X_{i'2}^2 \\ \sqrt{2} X_{i'1} X_{i'2} \end{pmatrix} \right\rangle \\
&= X_{i1}^2 X_{i'1}^2 + 2 X_{i1} X_{i2} X_{i'1} X_{i'2} + X_{i2}^2 X_{i'2}^2 \\
&= (X_{i1} X_{i'1} + X_{i2} X_{i'2})^2 \\
&= \langle X_i, X_{i'} \rangle^2 \\
&=: k(X_i, X_{i'})
\end{aligned}$$

Kernel-based learning methods

In summary,

$$\langle \Phi(X_i), \Phi(X_{i'}) \rangle = \langle X_i, X_{i'} \rangle^2 = k(X_i, X_{i'}).$$

In this context the function $k : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ is called **kernel**.

Accordingly, the SVM objective can be rewritten as follows:

$$\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \alpha_{i'} Y_i Y_{i'} k(X_i, X_i')$$

The objective remains convex if the **kernel** (Gram) **matrix**

$$\mathbf{K} = \left(k(X_i, X_{i'}) \right)_{1 \le i, i' \le n}$$

is (symmetric) positive semidefinite.

Kernel-based learning methods

Why not specifying the kernel function $k(\cdot, \cdot)$ instead of the feature map Φ?

Requirement on k:

there exists a map $\Phi : \mathcal{X} \to \mathcal{H}$ defined on the domain of inputs $\mathcal{X}$ mapping into an inner product space (Hilbert space) $\mathcal{H}$ with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ such that

$$k(x, x') = \left\langle \Phi(x), \Phi(x') \right\rangle_{\mathcal{H}} \quad \text{for all } x, x' \in \mathcal{X} \times \mathcal{X}.$$

Kernel-based learning methods

$$k(x, x') = \left\langle \Phi(x), \Phi(x') \right\rangle_{\mathcal{H}} \quad \text{for all } x, x' \in \mathcal{X} \times \mathcal{X}.$$

It turns out that the requirement is fulfilled if $k(\cdot, \cdot)$ is a (symmetric) **positive definite function**, i.e. if for any set of points $\{x_1, \ldots, x_m\} \subset \mathcal{X}$ the corresponding kernel Gram matrix

$$\mathbf{K} = \left(k(x_i, x_{i'}) \right)_{1 \leq i, i' \leq m}$$

is (symmetric) positive semidefinite.

We then call k a **positive definite kernel** or **Mercer kernel**.

Kernel-based learning methods

Approach:

map data in $\mathcal{X}$ to some (possibly ∞-dimensional) Hilbert space $\mathcal{H}$ via a suitable map Φ.

Then use a linear methods (linear regression, linear classification, ...) in $\mathcal{H}$.

The "kernel trick":

We do not have to perform the map explicitly because we know that inner products in $\mathcal{H}$ amount to evaluations of the corresponding kernel function k.

Kernel-based learning methods

Let k be a positive definite function on $\mathcal{X}$.

The **reproducing kernel Hilbert space** (RKHS) $\mathcal{H}_k$ associated with k is defined as the Hilbert space of functions obtained from

$$\text{span}\{k(x,\cdot),\ x \in \mathcal{X}\}$$

and inner product

$$\left\langle k(x,\cdot), k(x',\cdot) \right\rangle_{\mathcal{H}_k} := k(x,x').$$

(**reproducing property**).

Kernel-based learning methods

Given a positive definite kernel k on $\mathcal{X}$, the corresponding feature map Φ is hence given by

$$\begin{array}{rccl} \Phi: & \mathcal{X} & \to & \mathcal{H}_k \\ & x & \mapsto & \Phi(x) := k(x, \cdot) \end{array}$$

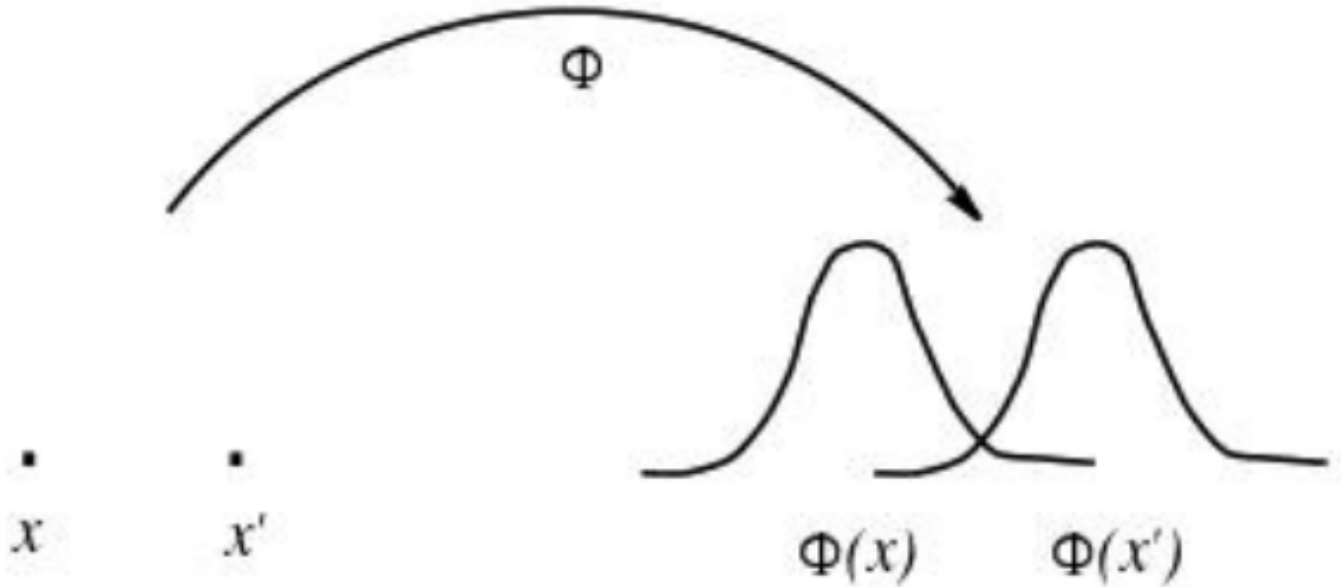
Intuition:

Think of $k(\cdot, \cdot)$ as a "similarity function" of pairs of points in $\mathcal{X}$.

Then we map a point x to a function $k(x, \cdot) \in \mathcal{H}_k$ that measures similarity with all other points in $\mathcal{X}$.

Kernel-based learning methods

Example: the Gaussian kernel



$$x \mapsto k(x, \cdot) = \exp(-\|\cdot - x\|_2^2/\sigma^2)$$

It can be shown that the Gaussian kernel is positive definite.
The associated space $\mathcal{H}_k$ is infinite-dimensional.

Kernel-based learning methods

Some kernel functions of interest:

$\mathcal{X} \subseteq \mathbb{R}^d$

1) Linear kernel

$$k(x, x') = \langle x, x' \rangle .$$

In this case, we do not do anything with the data: the corresponding feature map is given by $\Phi(x) = x$.

Kernel-based learning methods

Some kernel functions of interest:

$\mathcal{X} \subseteq \mathbb{R}^d$

2) Polynomial kernels with degree $\gamma \geq 1$:

$$\text{(inhomogeneous)} \quad k(x, x') = (1 + \langle x, x' \rangle)^\gamma,$$

$$\text{(homogeneous)} \quad \widetilde{k}(x, x') = \langle x, x' \rangle^\gamma .$$

The feature map corresponding to $\widetilde{k}$ extracts polynomial features of the form

$$x_1^{\gamma_1} \cdot \ldots \cdot x_d^{\gamma_d}, \;\; \gamma_1 \geq 0, \ldots, \gamma_d \geq 0, \;\; \sum_{j=1}^{d} \gamma_j = \gamma.$$

Kernel-based learning methods

The associated space $\mathcal{H}_{\widetilde{k}}$ has dimension $\binom{\gamma+d-1}{\gamma}$.

Suppose we have $d = 100$ features and let $\gamma = 3$. The associated space $\mathcal{H}_{\widetilde{k}}$ then already has dimension $D = 171,700$.

Because of the "kernel trick", we do need to perform the map Φ explicitly for the given data.

It suffices to operate in terms of the kernel on the original input domain.

Kernel-based learning methods

3) Translation-invariant kernels (or Fourier kernels).

$$k(x, x') = \kappa(x - x').$$

Most prominent representative is the Gaussian kernel (or RBF kernel):

$$\kappa(z) = \exp(-\|z\|_2^2/\sigma^2)$$

Remark:

Bochner's theorem ($\rightarrow$ Time Series Analysis, $\rightarrow$ Spatial Statistics) characterizes the set of continuous symmetric positive definite functions on $\mathbb{R}^d$.

In time series/spatial statistics, one typically speaks of "covariance function" (meaning the same as "kernel").

Kernel-based learning methods

The reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ associated with the Gaussian kernel is particularly "rich":

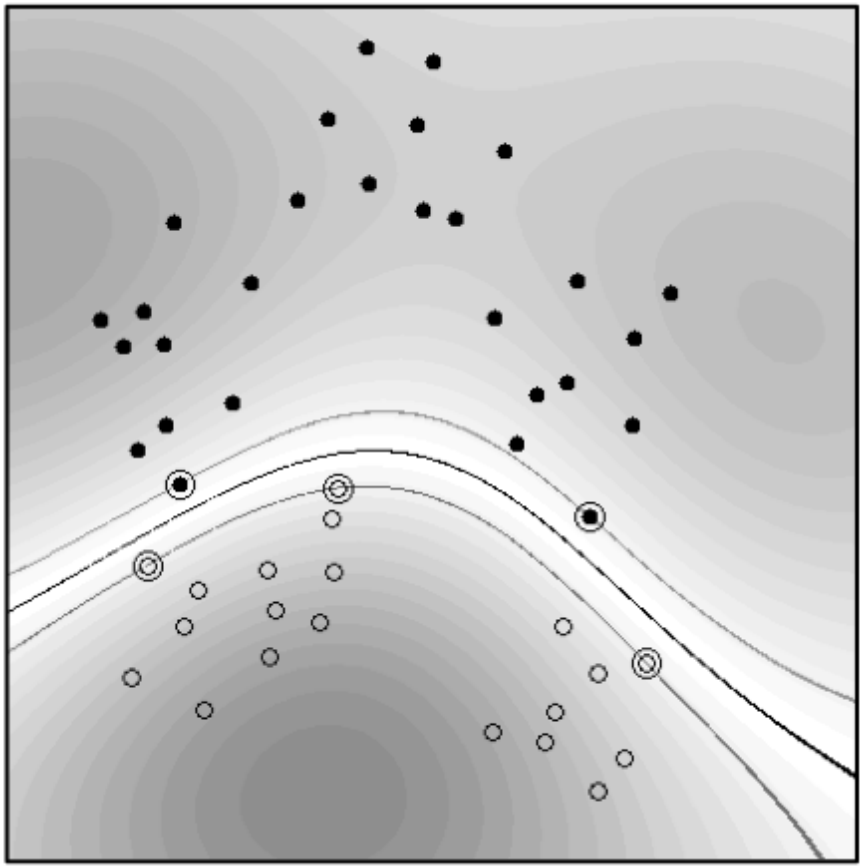
- it has infinite dimension,
- functions in $\mathcal{H}_k$ can be used to approximate continuous functions to arbitrary accuracy

This means:

- if we want to be flexible regarding the form a function to learn, the RBF kernel is a good choice
- at the same time: we need to be careful regarding overfitting.

Kernel-based learning methods

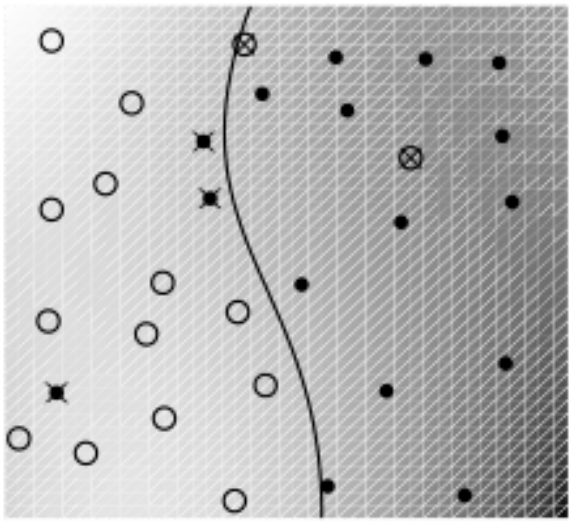
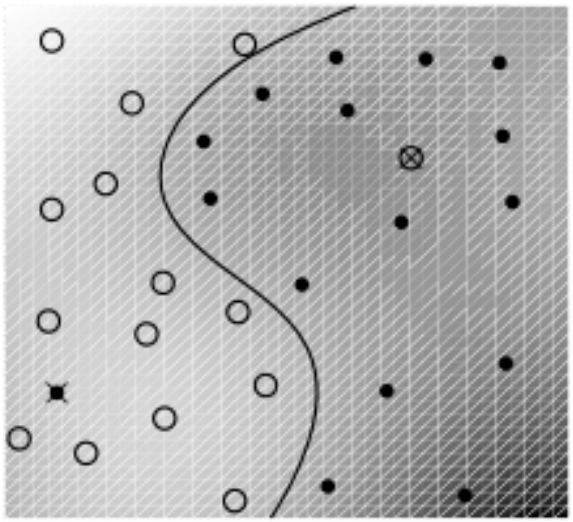
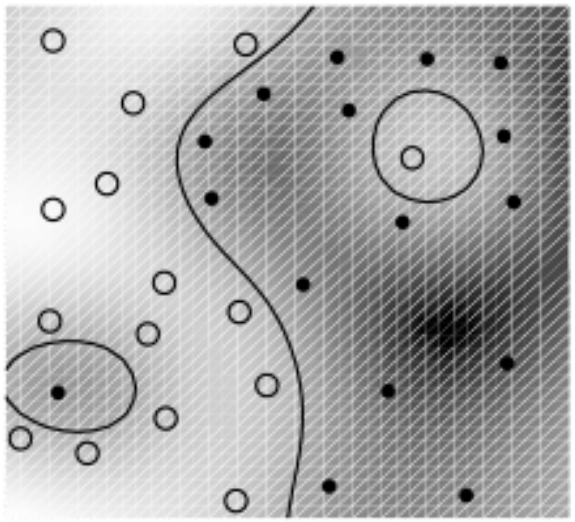
Illustration: support vector machines with the RBF kernel



Kernel-based learning methods

Influence of the choice of σ^2:

σ^2 decreasing from left to right.

Kernel-based learning methods

Rule of thumb for choosing σ:

between the 0.1 and 0.9 quantile of $\{\|X_i - X_{i'}\|_2\}_{i<i'}$

To be safe, one would choose σ based on cross-validation.

In general, we would like to use regularization to safeguard against overfitting.

How do we do regularized ERM with kernels?

Kernel-based learning methods

Regularized ERM over an RKHS:

Let us recall the paradigm of regularized empirical risk minimization:

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda \Omega(f).$$

where

- L is a loss function chosen depending on the learning problem,
- $\mathcal{F}$ is a class of functions ("hypothesis class"),
- $\Omega : \mathcal{F} \to \mathbb{R}_+$ is a regularizer.

Kernel-based learning methods

Regularized ERM over an RKHS:

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda \Omega(f).$$

Given a kernel k, we now consider this setup for

- $\mathcal{F} = \mathcal{H}_k$
- $\Omega(f) = \|f\|^2_{\mathcal{H}_k}$

What $\|f\|_{\mathcal{H}_k}$ eventually encodes depends on the kernel.

Kernel-based learning methods

Regularized ERM over an RKHS:

$$\min_{f \in \mathcal{H}_k} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda \| f \|_{\mathcal{H}_k}^2 .$$

For the RBF kernel (and others), $\mathcal{H}_k$ is ∞-dimensional.

That seems to imply that we cannot solve the above optimization problem.

It turns out, however, that the minimizer $\widehat{f}$ of the above problem is contained in a finite-dimensional subspace of $\mathcal{H}_k$ (to be specific: of dimension at most n).

Kernel-based learning methods

Representer Theorem (Kimeldorf and Wahba, 1971)

Consider the optimization problem

$$\min_{f \in \mathcal{H}_k} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda \|f\|^2_{\mathcal{H}_k}.$$

Then any minimizer $\widehat{f}(\cdot)$ of the above problem is of the form

$$\widehat{f}(\cdot) = \sum_{i=1}^{n} \alpha_i k(X_i, \cdot)$$

for coefficients $\{\alpha_i\}_{i=1}^n \subset \mathbb{R}$.

Kernel-based learning methods

Phrased differently, we have

$$\min_{f \in \mathcal{H}_k} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda \|f\|^2_{\mathcal{H}_k}$$

$$= \min_{f \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda \|f\|^2_{\mathcal{H}_k},$$

where $\mathcal{T} \subseteq \mathcal{H}_k$ is given by

$$\mathcal{T} = \left\{ f(\cdot) \in \mathcal{H}_k : \ f(\cdot) = \sum_{i=1}^{n} \alpha_i k(X_i, \cdot), \ \{\alpha_i\}_{i=1}^{n} \subset \mathbb{R} \right\}$$

Kernel-based learning methods

$$\min_{f \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda \|f\|^2_{\mathcal{H}_k},$$

$$\mathcal{T} = \left\{ f(\cdot) \in \mathcal{H}_k : \ f(\cdot) = \sum_{i=1}^{n} \alpha_i k(X_i, \cdot), \ \{\alpha_i\}_{i=1}^{n} \subset \mathbb{R} \right\}$$

This implies

1)

$$f(X_i) = \sum_{j=1}^{n} \alpha_j k(X_j, X_i)$$

Kernel-based learning methods

$$\min_{f \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda \|f\|^2_{\mathcal{H}_k},$$

$$\mathcal{T} = \left\{ f(\cdot) \in \mathcal{H}_k : \; f(\cdot) = \sum_{i=1}^{n} \alpha_i k(X_i, \cdot), \; \{\alpha_i\}_{i=1}^{n} \subset \mathbb{R} \right\}$$

2)

$$\begin{aligned} \|f\|^2_{\mathcal{H}_k} &= \langle f, f \rangle_{\mathcal{H}_k} \\ &= \left\langle \sum_{i=1}^{n} \alpha_i k(X_i, \cdot), \sum_{i=1}^{n} \alpha_i k(X_i, \cdot) \right\rangle_{\mathcal{H}_k} \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(X_i, X_j) \end{aligned}$$

using the **reproducing property**: $\langle k(x, \cdot), k(x', \cdot) \rangle_{\mathcal{H}_k} = k(x, x')$.

Kernel-based learning methods

Combining 1) and 2), we have (with slight abuse of notation)

$$\min_{f \in \mathcal{T}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) + \lambda \| f \|^2_{\mathcal{H}_k}$$
$$= \min_{\alpha \in \mathbb{R}^n} \frac{1}{n} L(\mathbf{y}, K\alpha) + \lambda \alpha^\top \mathbf{K} \alpha.$$

Hence, we only need the kernel Gram matrix $K = (k(x_i, x_{i'}))_{1 \le i, i' \le n}$ of the training data.

With that, we find an optimal set of coefficients $\{\alpha_i\}_{i=1}^n$.

Kernel-based learning methods

Kernel Ridge Regression (KRR):

Specializing to squared loss, we obtain the kernel ridge regression problem:

$$\min_{\alpha \in \mathbb{R}^n} \frac{1}{n} \|\mathbf{y} - K\alpha\|_2^2 + \lambda \alpha^\top K \alpha.$$

and an optimal set of coefficients is given by

$$\widehat{\alpha} = (n\lambda I + K)^{-1}\mathbf{y}.$$

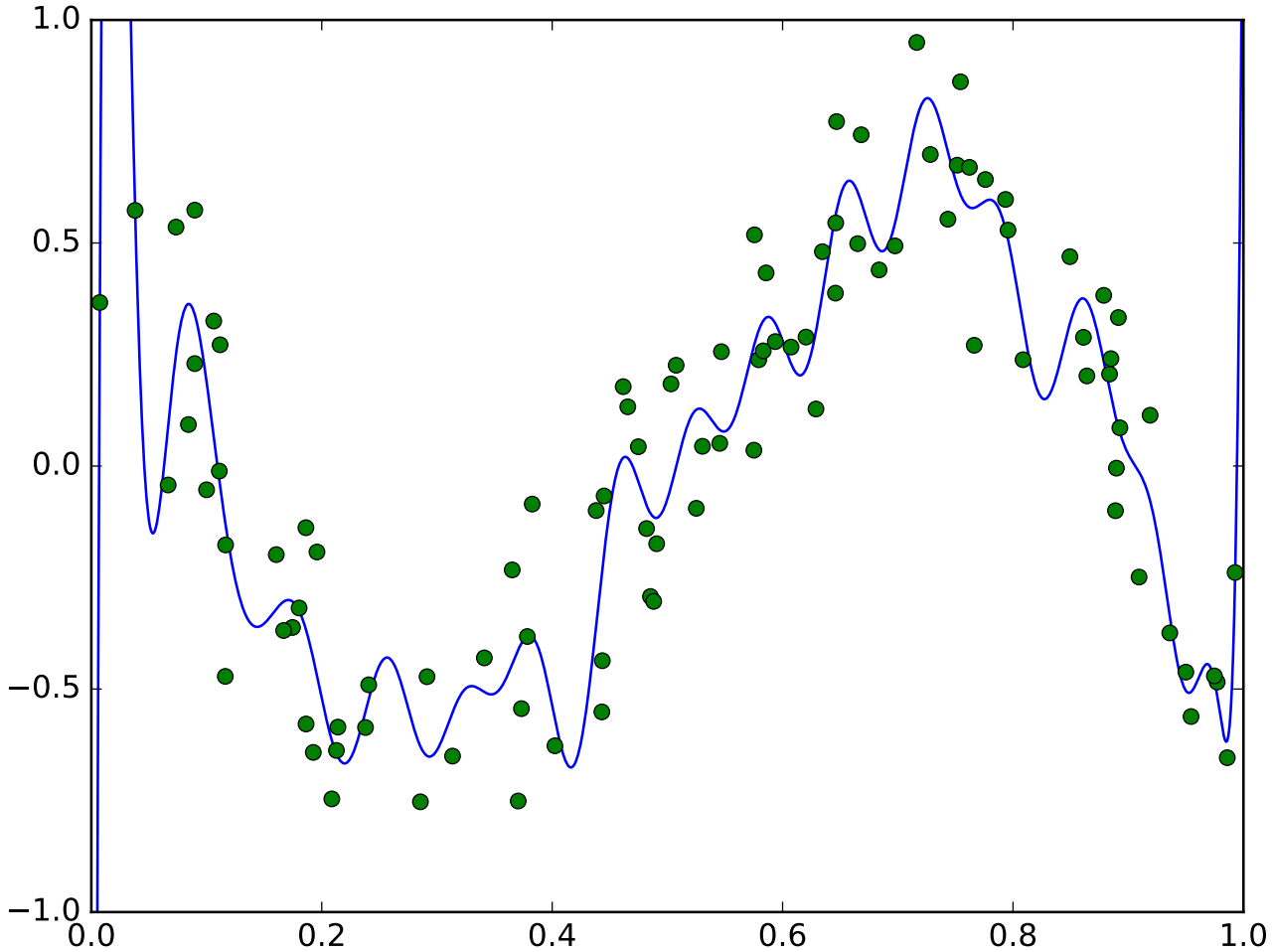
The predictions on the training set result as

$$\begin{aligned} K\widehat{\alpha} &= K(n\lambda I + K)^{-1}\mathbf{y} \\ &= A_\lambda \mathbf{y} \end{aligned}$$

$\Rightarrow$ we can use Mallow's C_p for selecting λ.

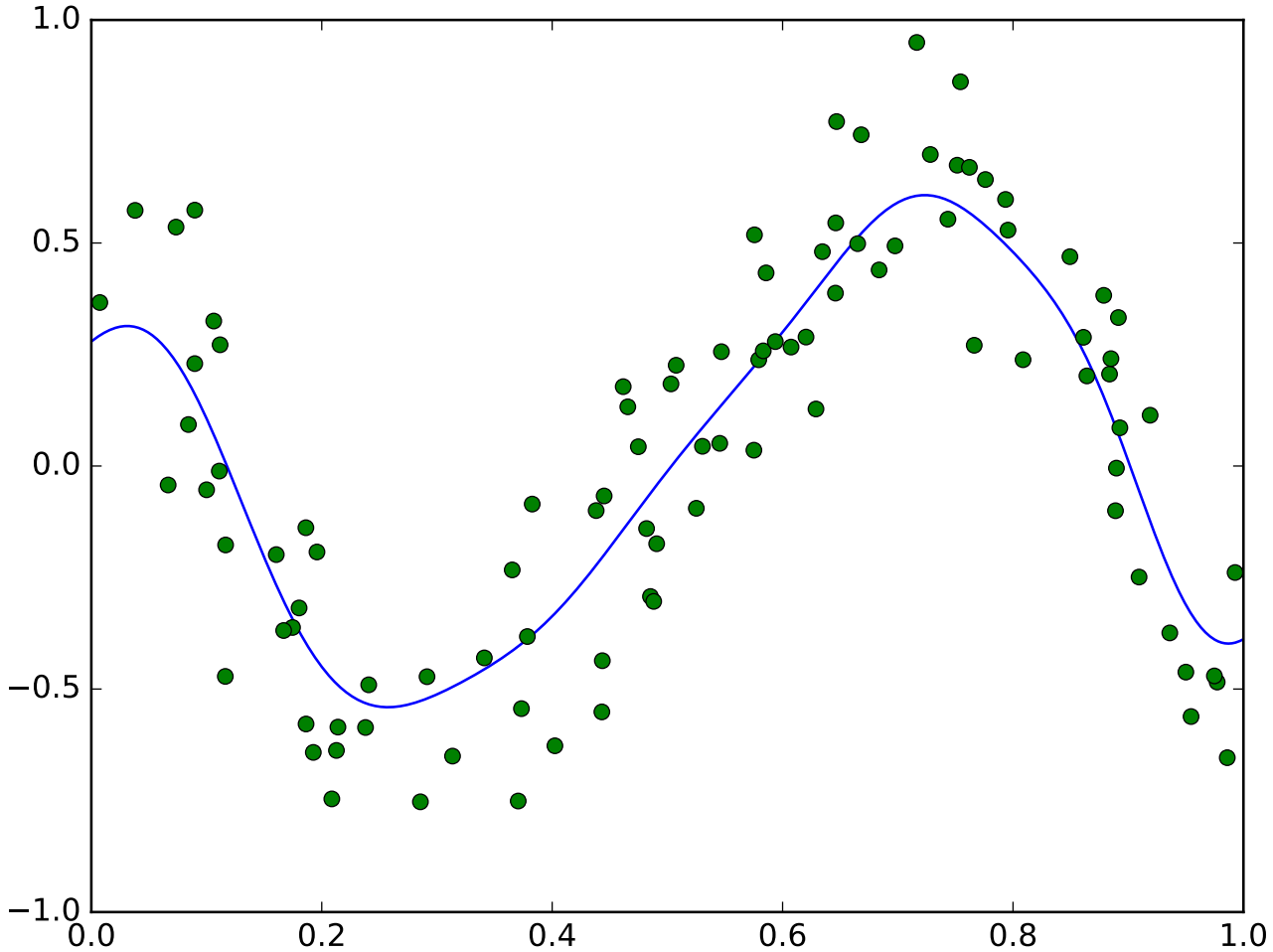
Kernel-based learning methods

KRR, $\sigma = 0.01$, $\lambda = 10^{-10}$.



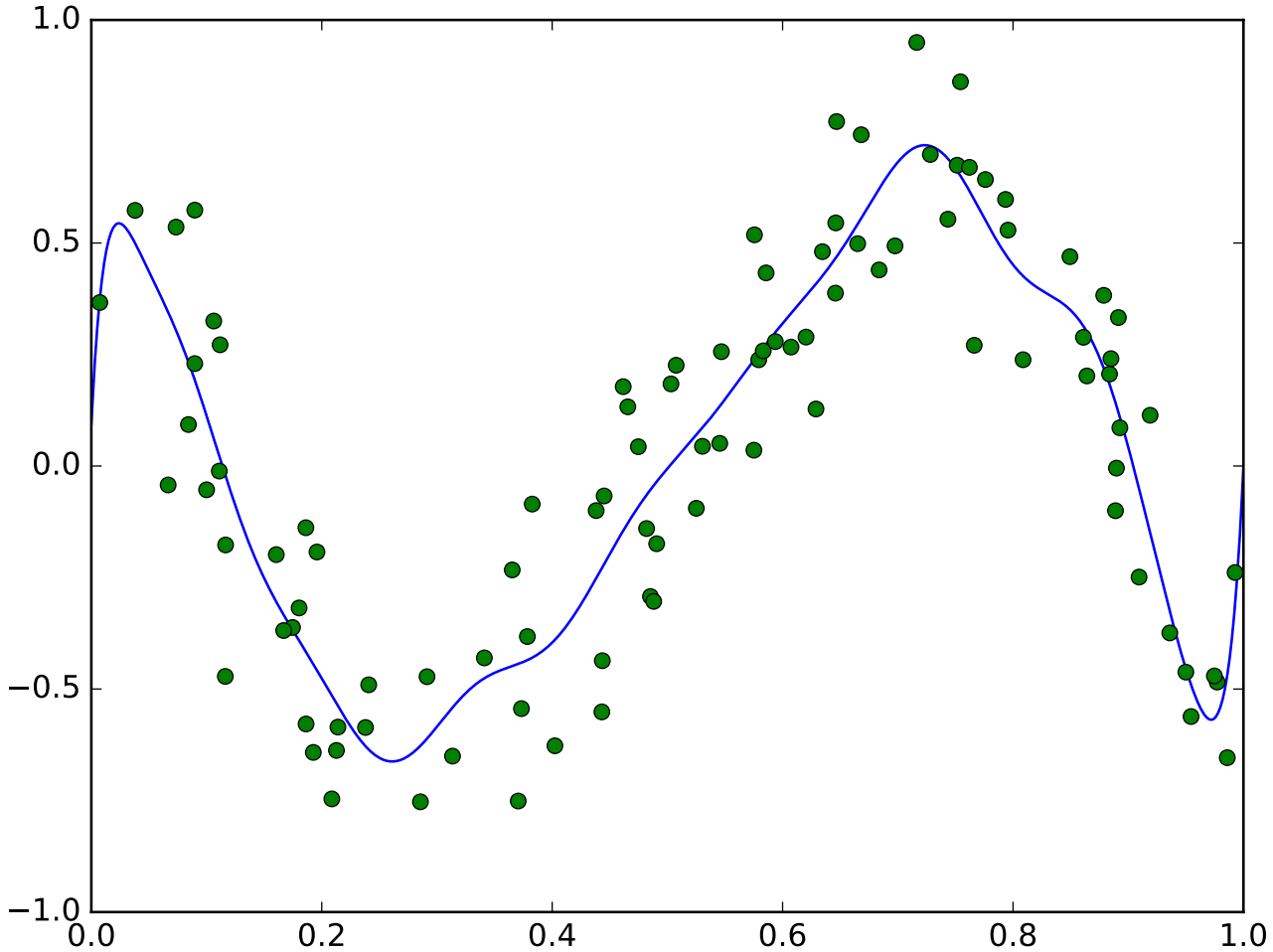
Kernel-based learning methods

$\sigma = 0.01$, $\lambda = 1.27$.



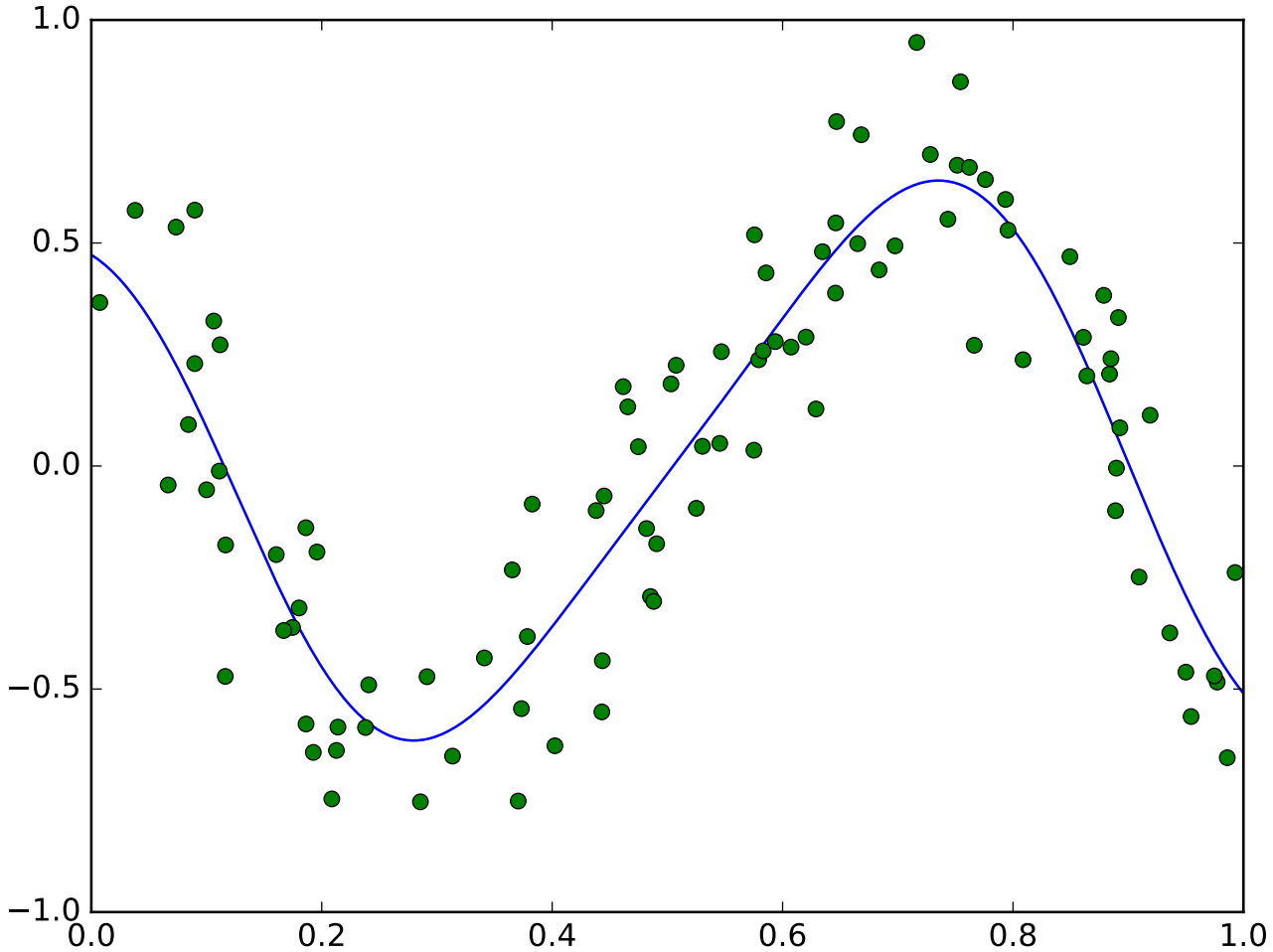
Kernel-based learning methods

$\sigma = 0.05$, $\lambda = 10^{-10}$.



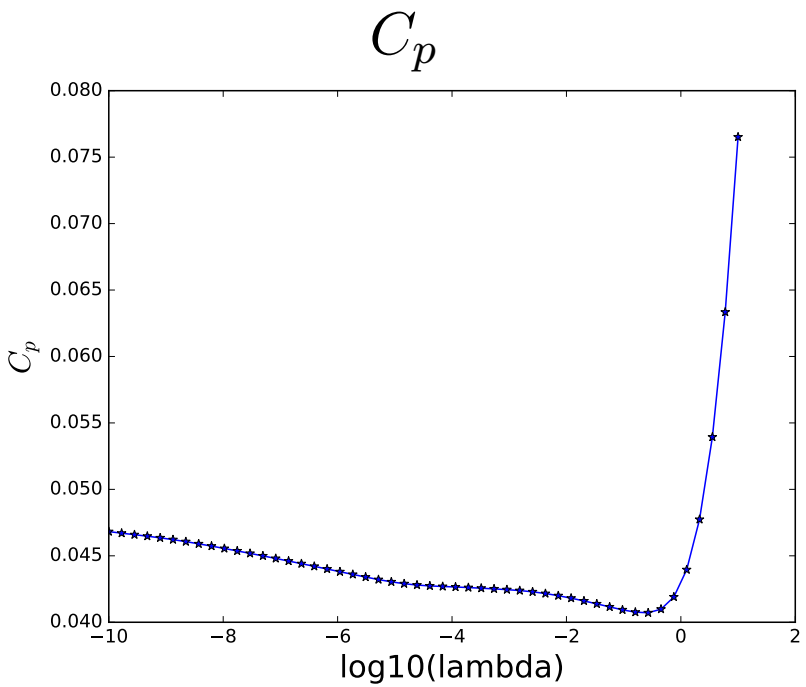
Kernel-based learning methods

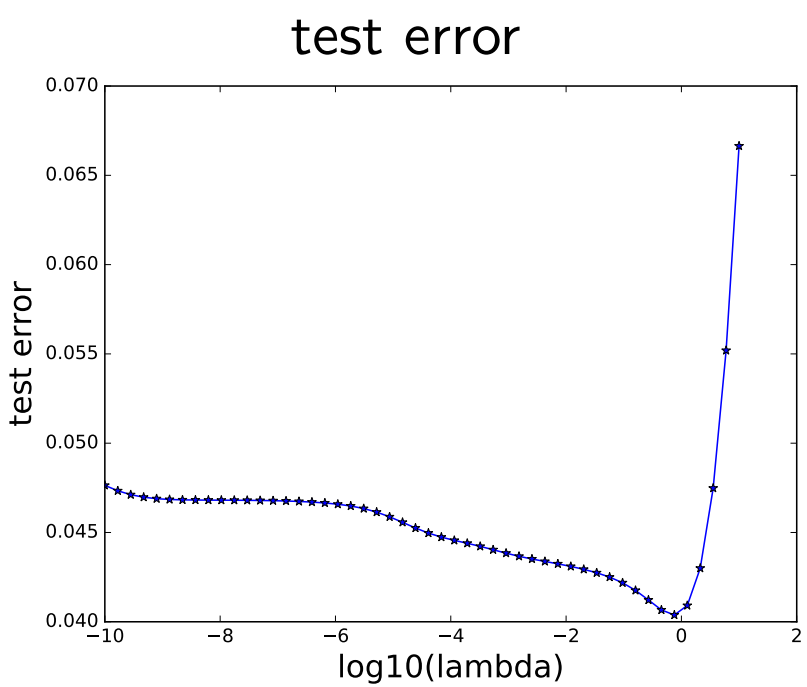
$\sigma = 0.05$, $\lambda = 0.27$.



Kernel-based learning methods

Choice of λ via Mallow's C_p:





Kernel-based learning methods

Kernel Logistic Regression:

In view of the representer theorem, we need to solve

$$\min_{\alpha \in \mathbb{R}^n} \frac{1}{n} L(\mathbf{y}, K\alpha) + \lambda \alpha^\top K \alpha.$$

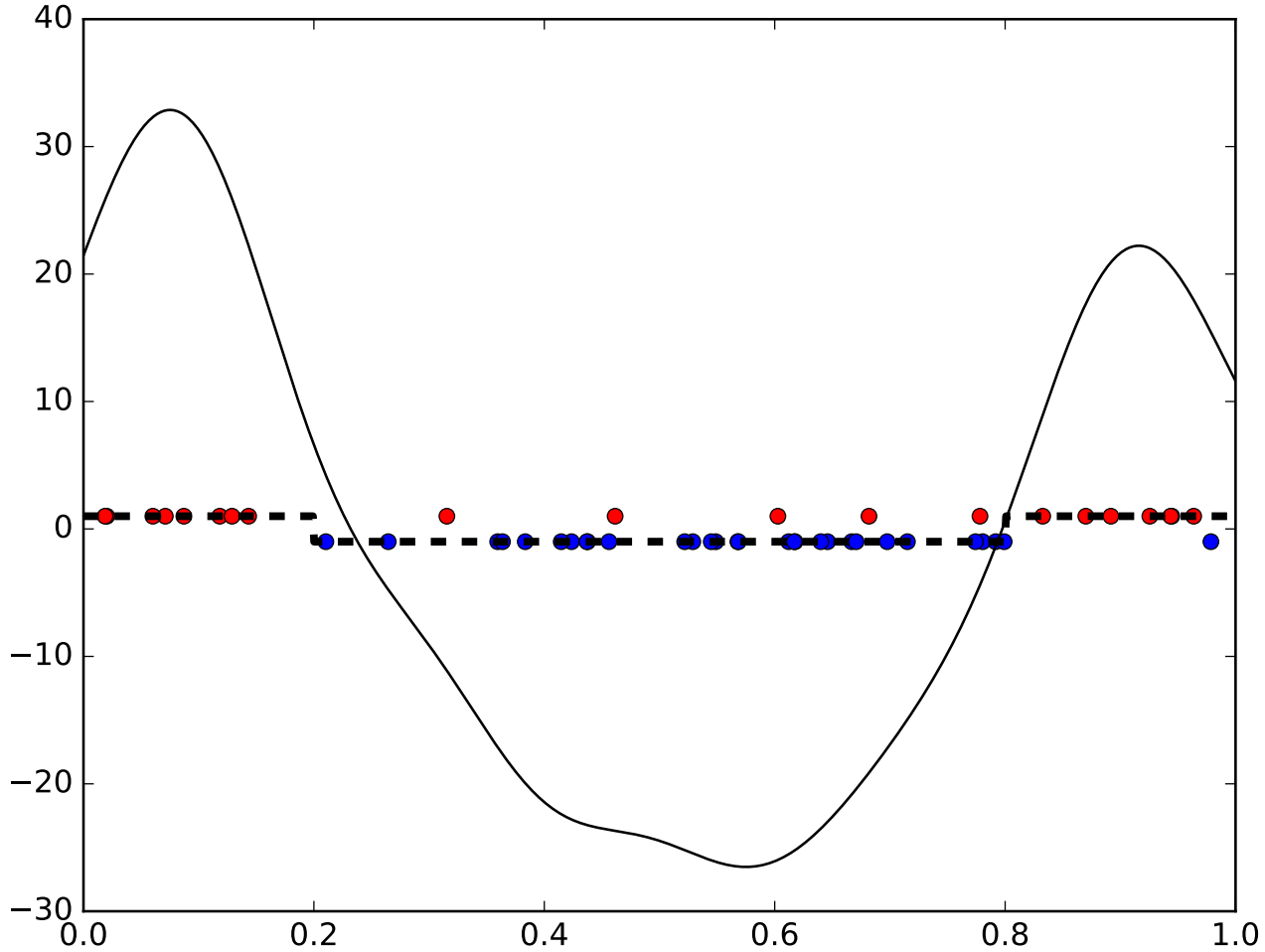
where L is the logistic loss. This yields

$$\min_{\alpha \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^n \log(1 + \exp(-Y_i (K\alpha)_i)) + \lambda \alpha^\top K \alpha.$$

$$= \min_{\alpha \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^n \log \left(1 + \exp \left(-Y_i \sum_{j=1}^n \alpha_j k(X_i, X_j) \right) \right) + \lambda \alpha^\top K \alpha.$$

Kernel-based learning methods

Application to the problem in HW2 ($n = 50$, RBF kernel, $\sigma = 0.01$, $\lambda = .01$):



Kernel-based learning methods

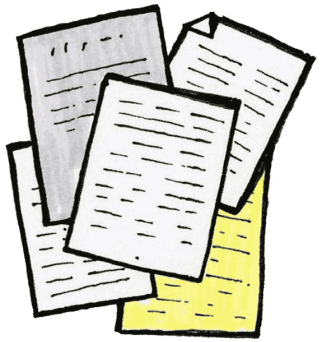
"Kernelization": any data analysis tool only depending on inner products of the data points can be "kernelized" by replacing inner products with kernels.

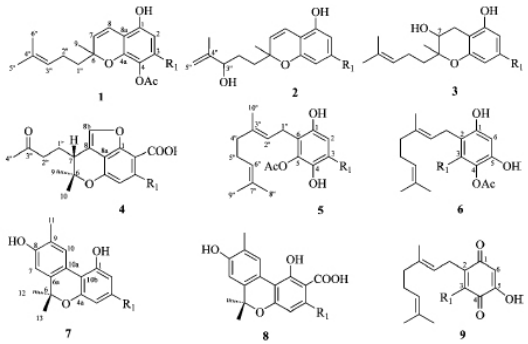
- kernel LDA,
- kernel PCA,
- kernel canonical correlation analysis,
- kernel k-means,
- kernel partial least squares,
- kernel statistical tests,
- $\vdots$

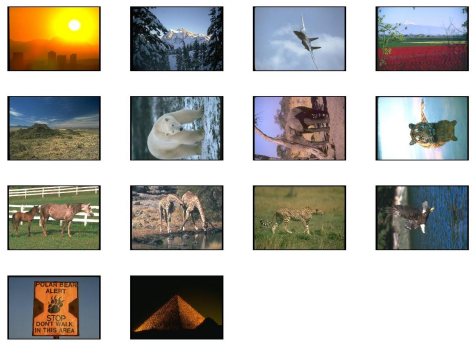
Kernel-based learning methods

Kernels on non-Euclidean domains:

A second important benefit of the kernel framework is its applicability in the case where $\mathcal{X}$ is *not* a subset of $\mathbb{R}^d$.







Kernel-based learning methods

As long as we can define a suitable kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (for *arbitrary* $\mathcal{X}$), we can use our favorite (kerneliz-able) method for classification, regression etc.

The choice of the kernel is critical:

- it needs to be an appropriate measure of similarity on the domain of interest
- it needs to be efficiently computable

Achieving both at the same time can be a challenge for complex objects such e.g. labelled graphs.

Kernel-based learning methods

One basic approach is based on histograms.

For text documents, one counts k-grams:

Full sentence	It does not, however, control whether an exaction is within Congress's power to tax.
Unigrams	"It"; "does"; "not,"; "however,"; "control"; "whether"; "an"; "exaction"; "is"; "within"; "Congress's"; "power"; "to"; "tax."
Bigrams	"It does"; "does not,"; "not, however,"; "however, control"; "control whether"; "whether an"; "an exaction"; "exaction is"; "is within"; "within Congress's"; "Congress's power"; "power to"; "to tax."
Trigrams	"It does not"; "does not, however"; "not, however, control"; "however, control whether"; "control whether an"; "whether an exaction"; "an exaction is"; "exaction is within"; "is within Congress's"; "within Congress's power"; "Congress's power to"; "power to tax."

A k-gram is a continguous sequence of k-words.

Kernel-based learning methods

Similar ideas can be employed for strings (histograms over substrings or subsequences) or images (color histograms).

A simple kernel on histograms is the resemblance kernel (also known as Jaccard similarity):

$$k(A, B) = \frac{|A \cap B|}{|A \cup B|},$$

where A and B are two sets (allowing duplicate elements).

Kernel-based learning methods

A serious concern about kernel-based is their scalability to massive data sets.

- evaluation and storage of the kernel matrix has complexity $O(n^2)$.
- running a kernel-based algorithm such as regression or PCA has a runtime of $O(n^3)$.

Various strategies have been proposed to deal with this issue; it remains an area of active research.